Probe Scheduling for Efficient Detection of Silent Failures

Edith Cohen [*‡] Avinatan Hassidim [†¶] Haim Kaplan [‡] Yishay Mansour [‡]
Danny Raz [§¶] Yoav Tzur [¶]

edith@cohenwang.com, {avinatan,razdan,yoavtz}@google.com,
{haimk,mansour}@cs.tau.ac.il

Abstract

Most discovery systems for silent failures work in two phases: a continuous monitoring phase that detects presence of failures and a localization phase that identifies the faulty element(s). This separation is important because localization requires more significant resources than detection and should be initiated only when a fault is present.

Detection, which we focus on, uses a schedule of probe packets. We propose ways to improve the system efficiency by addressing both the selection of appropriate objectives for detection time, based on costs associated with particular failures, and the efficient design of schedules, which satisfy objectives with minimum probing overhead. Our work unifies the treatment of SUM and MAX objectives and of stochastic and deterministic schedules and allows us to compare and relate these alternatives.

We define *memoryless schedules* – a subclass of stochastic schedules which is simple and suitable for distributed deployment. We show that optimal memorlyess schedules can be efficiently computed by convex programs (for SUM objectives) or linear programs (for MAX objectives), and surprisingly perhaps, are guaranteed to have expected detection times that are not too far off the (NP hard) stochastic optima.

Deterministic schedules allow us to bound the maximum (rather than expected) cost of undetected faults, but like stochastic schedules, are NP hard to optimize. We propose novel efficient deterministic schedulers with analytic performance guarantees and simulate them on real networks. By relating performance to the computed memoryless optima, we can see that on many instances, our deterministic schedules are nearly optimal.

[*]Microsoft Research (CA, USA)
[†]Bar Ilan University (Israel)
[‡]Tel Aviv University (Tel Aviv, Israel)
[§]Technion (Israel)
[¶]Google, Inc. Israel R&D Center

1 Introduction

Prompt detection of failures of network elements is a critical component of maintaining a reliable network. Silent failures, which are not announced by the failed elements, are particularly challenging and can only be discovered by active monitoring.

Failure identification systems [10, 11, 13, 12] typically work in two phases: First detecting presence of a fault and then localizing it. The rational behind this design is that detection is an easier problem than localization and requires light weight mechanisms that have little impact on network performance. Once the presence of a fault is confirmed, more extensive tools which may consume more resources are deployed for localizing the fault. Moreover, in some cases, it is possible to bypass the problem, by rerouting through a different path, quicker than the time it takes to pinpoint or correct the troubled component.

A lightweight failure detection mechanism, which relies on the existing infrastructure, uses probe packets or *probes* that are sent from certain hosts or between OD pairs along the existing routing infrastructure (see Figure 1). The elements we monitor can be physical links [10], combination of components and paths [11], or logical components of network elements like the forwarding rules in the switches of a software-defined network [12]. If one of the elements on the probe path fails, the probing packet will not reach the destination, and in this case the probe has detected a failure.

The goal is to design schedules which optimize the tradeoff between failure detection time or more generally, the cost or expected cost associated with failures, and the probing overhead. We work with continuous testing where failures may occur at any time during the (ongoing) monitoring process, and we would like to detect the failure soon after it occurs. This is in contrast to *one-time testing*, where tests starts at some point in time and the goal is to detect the presence of an existing failure. Continuous monitoring comes in many flavors: deployment can be centralized or distributed across the network and may require following a fixed sequence of probes (*deterministic* schedules) or allow for randomization (*stochastic* schedules). There are also several natural objectives: MAX_e objectives set a different detection time target for each element and aim to meet all these targets with minimum overhead whereas SUM_e objectives aim to minimize a weighted sum (average) of detection times.

We unify the treatment of these diverse methods and objectives in a common framework which we then use to develop scheduling algorithms and to study performance – whilst stronger objectives are often desirable, it is important to quantify their increased cost.

We define a simple and appealing class of stochastic schedules which we call *memoryless schedules*. Memoryless schedules perform continuous testing by invoking tests selected independently at random according to some fixed distribution. The stateless nature of memoryless scheduling translates to minimum deployment overhead and also makes them very suitable in distributed settings, where each type of test is initiated by a different controller. We show that the optimization problem of computing the probing frequencies under which a memoryless schedule optimizes a SUM_e objective can be formulated as a convex program and when optimizing MAX_e objectives, as a linear program. In both cases, with respect to either SUM_e or MAX_e objectives, the optimal memoryless schedule can be computed efficiently. This is in contrast to general stochastic schedules, over which we show that the optima are NP-hard to compute. Surprisingly perhaps, we also show that the natural and efficiently optimizable memoryless schedules have expected detection times that are guaranteed to be within a factor of two from the respective optimal stochastic schedule of the same objective. Moreover, detection times are geometrically distributed, and therefore variance in detection time is well-understood, which is not necessarily so for general stochastic schedules.

Our convex program formulation for the optimal probing frequencies for SUM_e objectives generalizes Kleinrock's classic "square-root law" to resource allocation problems with *subset tests*. Kleinrock's law, which applies to the special case of *singleton tests* (where each test can detect the failure of a single element[1]), states that the optimal probing frequencies are proportional to the square root of the weighting frequencies [9].

Applications requiring hard guarantees on detection times or failure costs or implementations requiring a fixed schedule prompt us to consider *deterministic scheduling*. Deterministic schedulers, however, are less suitable for distributed deployment and also come with an additional cost: the optimum of an objective on a deterministic schedule can exceed the expectation of the same objective over stochastic schedules. We study the inherent gap (which we call the D2M gap) between these optima. We show that for deterministic scheduling, performance within SUM_e or MAX_e objectives further depends on the dependence on time (average or maximum). While all variants are NP hard, there is significant variation between attainable approximation ratios for the different objectives: The stricter MAX_e and SUM_e objectives, requiring good performance at any point in time, can not be approximated better than logarithmic factors whereas the weakest, which consider average performance over time, have factor 2 approximations.

Building on this, we efficiently construct deterministic schedules with approximation and D2M ratios that meet the analytic bounds. Our *random tree* (R-Tree) schedulers derive a deterministic schedule from the probing frequencies of a memoryless schedule, effectively "derandomizing" the schedule while attempting to loose as little as possible on the objective in the process. We show that when seeded, respectively, with a SUM_e or MAX_e optimal memoryless schedule, we obtain deterministic schedules with approximation ratio of $O(\log \ell)$ for the strongest SUM_e objective and ratio $O(\log \ell + \log n)$ for the strongest MAX_e objective, where n is the number of elements and ℓ is the maximum number of tests that can detect the failure of a particular element. We also present the Kuhn-Tucker (KT) scheduler which is guided by the Kuhn Tucker conditions on the convex program computing the optimal SUM_e memoryless schedules and is geared to SUM_e objectives. The KT scheduler adapts gracefullt to changing priorities, for example in response to change traffic patterns in the network, and allows for a seamless transition.

Finally, we evaluate the different schedulers on realistic networks of two different scales: We use both a globe-spanning backbone network and a folded-Clos network, which models a common data center architecture. In both cases, the elements we are testing are the network links. For the backbone, our tests are the set of MPLS paths and for the Clos network we use all routing paths. We demonstrate how our suite of schedulers offers both strong analytic guarantees, good performance, and provides a unified view on attainable performance with respect to different objectives. We also demonstrate how our theoretical analysis explains observed performance and supports educated further tuning of schedulers.

An important contribution of our work is the unified general treatment of multiple settings and objectives which were studied singularly in previous work, and offering a precise understanding of their relations and tradeoffs. Our work facilitates an informed choice of the proper objective for the problem at hand and efficient algorithms to compute or approximate the optimal solution.

The paper is structured as follows. In Section 2 we present our model, general stochastic and deterministic schedules, and explain the different objectives. Memoryless schedules are introduced in Section 3. Deterministic scheduling is discussed Section 4, followed by the R-Tree scheduler in Section 5 and Kuhn-Tucker schedulers in Section 6. Experimental results are presented in Section 8, extension of the model to probabilistic tests in discussed in Section 7, and related work is discussed in Section 9.

2 Model

An instance of a *test scheduling* problem is specified by a set V of elements (which can be thought of as network elements or links) of size n with a weight function $\boldsymbol{p}$ (which can be thought of as priority or importance of the elements) and a set $\mathcal{S}$ of tests (probe paths) of size m. For $i \in [m]$, test i is specified by

[1]This also works for sets of non-overlapping elements.

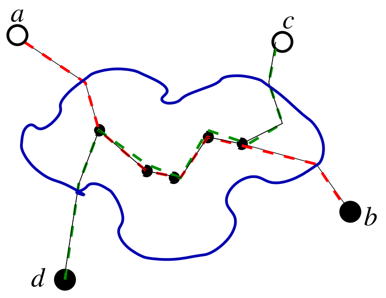


Figure 1: Network and elements covered by ab and cd origin-destination tests.

a subset $s_i \subset V$ of elements. A failure of element e can be detected by probing i if and only if $e \in s_i$, that is, if and only if test i contains the failed element.[2] We use ℓ_e to indicate the number of tests which include element e and $\ell \equiv \max_e \ell_e$.

Continuous testing is specified by *schedules* which generate an infinite sequence $\boldsymbol{\sigma}$ of tests. The schedule can be *deterministic* (where it is just a finite cycle of tests) or *stochastic*, in which case, it is defined as a distribution over a countable set of deterministic sequences. Note that this general definition covers also *adaptive* stochastic schedules in which the probability distribution of the tests at time t depends on the actual tests preformed prior to time t. We also introduce *memoryless* schedules, which are a special subclass of stochastic schedules, in which the probability distribution of the tests is fixed over time.

2.1 Objectives

Objectives for a testing schedule aim to minimize a certain function of the number of tests invoked until a fault is detected. (We essentially measure time passed until the fault is detected by the "number of probes" required to discover it If the probing rate is fixed this is indeed the time.) Several different natural objectives had been considered in the literature. Here we consider all these objectives through a unified treatment which allows us to understand how they relate to each other and how they can be computed or approximated.

The *detection time* $\mathrm{T}_{\boldsymbol{\sigma}}(e,t)$ for element e at time t by a schedule $\boldsymbol{\sigma}$ is the expected time to detect a fault to element e that occurs at time t. If the schedule is deterministic, then $\mathrm{T}_{\boldsymbol{\sigma}}(e,t) = \min_{h\geq 0} e \in s_{\sigma_{h+t}}$. If the schedule is stochastic, we take the expectation over sequences

$$\mathrm{T}_{\boldsymbol{\sigma}}(e,t) = \mathsf{E}_{\boldsymbol{\sigma}}[\min_{h\geq 0} e \in s_{\sigma_{h+t}}] \ .$$

We classify natural objectives as *MAX*$_e$, when aiming to minimize the (weighted) maximum over elements, or as *SUM*$_e$ when aiming to minimize a weighted sum over elements. Both types of objectives are defined with respect to the weight function $\boldsymbol{p}$ over elements.

The weighting, or priorities of different elements, can capture the relative criticality of the element which in turn, can be set according to the volume or quality of service level of the traffic they handle. With the SUM$_e$ objectives, the weights can also correspond to estimated probability that elements fail, in which case the weighted objective capture the expected detection time after a failure, or to the product of failure probability of the element and cost of failure of this element, in which case the weighted objective is the expected cost of a failure. With the MAX$_e$ objectives we can use $p_e \equiv 1/\tau_e$, where τ_e is the minimum desired detection time for a failure of element e, or the cost of a unit of downtime of element e. We then aim to minimize the maximum cost of a failing element. In the sequel we assume that weights are scaled so that with SUM$_e$, $\sum_e p_e = 1$, and with MAX$_e$, $\max_e p_e = 1$.

[2]This can be extended to the case where failures are detected with some positive probability.

We further distinguish between objectives based on their dependence on time: average or maximum. To facilitate treatment of the different objectives we define the operators M_e and E_e, which perform weighted maximum or average over elements, and M_t and E_t, which perform maximum or average over time. More precisely, for a function g of time or a function f over elements:

$$\begin{aligned}\mathsf{M}_t[g] &= \sup_{t\geq 1} g(t)\\ \mathsf{E}_t[g] &= \lim_{h\to\infty}\frac{\sum_{t=1}^{h} g(t)}{h}\\ \mathsf{M}_e[f] &= \max_e p_e f(e)\\ \mathsf{E}_e[f] &= \sum_e p_e f(e)\end{aligned}$$

An application of the operator E_t requires that the limit exists and M_t requires that $g(t)$ is bounded.

When the operators are applied to the function $T(e,t)$, we use the shorthand $\mathsf{M}_t[e|\boldsymbol{\sigma}] \equiv \mathsf{M}_t[\mathrm{T}_{\boldsymbol{\sigma}}(e,t)]$, $\mathsf{E}_t[e|\boldsymbol{\sigma}] \equiv \mathsf{E}_t[\mathrm{T}_{\boldsymbol{\sigma}}(e,t)]$, $\mathsf{M}_e[t|\boldsymbol{\sigma}] \equiv \mathsf{M}_e[\mathrm{T}_{\boldsymbol{\sigma}}(e,t)]$, $\mathsf{E}_e[t|\boldsymbol{\sigma}] \equiv \mathsf{E}_e[\mathrm{T}_{\boldsymbol{\sigma}}(e,t)]$. For a particular element e, $\mathsf{M}_t[e|\boldsymbol{\sigma}]$ is the maximum over time t of the expected (over sequences) number of probes needed to detect a failure of e that occurred in time t, and $\mathsf{E}_t[e|\boldsymbol{\sigma}]$ is the expectation (over sequences) of the (limit of) the average over time t of the number of probes needed to detect a failure of e that occurred in time t. For a particular time t, $\mathsf{M}_e[t|\boldsymbol{\sigma}]$ is the (weighted) maximum over the elements of the expected detection time of a failure at t, and $\mathsf{E}_e[t|\boldsymbol{\sigma}]$ is the weighted sum over the elements of their expected detection times at t. We consider all objectives that we can obtain from combinations of these operators. The operator pairs M_e and M_t (maximum over time or over elments) and E_e and E_t (average of expectation) commute, but other pairs do not, and we obtain six natural objectives, three MAX$_e$ and three SUM$_e$.

MAX$_e$ objectives: In order of decreasing strictness, the three MAX$_e$ objectives are $\mathsf{M}_e[\mathsf{M}_t[e|\boldsymbol{\sigma}]]$, the weighted maximum over elements of the maximum over time of the detection time. In this objective and the following ones $\mathsf{M}_e[\mathsf{E}_t[e|\boldsymbol{\sigma}]]$, the weighted maximum over elements of the average over time, and $\mathsf{E}_t[\mathsf{M}_e[t|\boldsymbol{\sigma}]]$, the average over time of the maximum element at that time. We shorten notation as follows.

$$\begin{aligned}\mathsf{M}_e\mathsf{M}_t[\boldsymbol{\sigma}] &= \mathsf{M}_e[\mathsf{M}_t[e|\boldsymbol{\sigma}]] \equiv \sup_{e,t} p_e \mathrm{T}_{\boldsymbol{\sigma}}(e,t)\\ \mathsf{M}_e\mathsf{E}_t[\boldsymbol{\sigma}] &= \mathsf{M}_e[\mathsf{E}_t[e|\boldsymbol{\sigma}]] \equiv \max_e p_e \mathsf{E}_t[e|\boldsymbol{\sigma}]\\ \mathsf{E}_t\mathsf{M}_e[\boldsymbol{\sigma}] &= \mathsf{E}_t[\mathsf{M}_e[t|\boldsymbol{\sigma}]] \equiv \lim_{h\to\infty}\frac{1}{h}\sum_{t=1}^{h}\max_e p_e \mathrm{T}_{\boldsymbol{\sigma}}(e,t)\ . \qquad (1)\end{aligned}$$

SUM$_e$ objectives: In order of decreasing strictness, the three SUM$_e$ objectives are $\mathsf{E}_e[\mathsf{M}_t[e|\boldsymbol{\sigma}]]$, the weighted sum over elements e of the maximum over time t of detection time $T(e,t)$, $\mathsf{M}_t[\mathsf{E}_e[t|\boldsymbol{\sigma}]]$, the maximum over time of the weighted sum over e, and $\mathsf{E}_e[\mathsf{E}_t[e|\boldsymbol{\sigma}]]$, the weighted sum over elements of the average over time. We similarly shorten notation as follows.

$$\begin{aligned}\mathsf{E}_e\mathsf{M}_t[\boldsymbol{\sigma}] &= \mathsf{E}_e[\mathsf{M}_t[e|\boldsymbol{\sigma}]] = \sum_e p_e \mathsf{M}_t[e|\boldsymbol{\sigma}]\\ \mathsf{M}_t\mathsf{E}_e[\boldsymbol{\sigma}] &= \mathsf{M}_t[\mathsf{E}_e[t|\boldsymbol{\sigma}]] = \sup_t \sum_e p_e \mathrm{T}_{\boldsymbol{\sigma}}(e,t) = \sup_t \mathsf{E}_e[t|\boldsymbol{\sigma}]\\ \mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] &= \mathsf{E}_e[\mathsf{E}_t[e|\boldsymbol{\sigma}]] = \sum_e p_e \mathsf{E}_t[e|\boldsymbol{\sigma}]\end{aligned}$$

When the schedule $\boldsymbol{\sigma}$ is clear from context, we omit the reference to it in the notation. There clearly exist inifinite sequences of probes, deterministic or stochastic, over which our objectives are not defined. The $\mathsf{M}_e\mathsf{M}_t$, $\mathsf{E}_e\mathsf{M}_t$, and $\mathsf{M}_t\mathsf{E}_e$ are defined when $\mathsf{M}_t[e]$ is defined for all elements and the $\mathsf{M}_e\mathsf{E}_t$ and $\mathsf{E}_e\mathsf{E}_t$ when $\mathsf{E}_t[e]$ is defined for all elements. The $\mathsf{E}_t\mathsf{M}_e$ requires that the limit in Equation (1) exists. Formally, we define a schedule to be *valid* if for all elements e, $\mathsf{M}_t[e]$ and $\mathsf{E}_t[e]$ are well defined, and for all tests i the relative frequency of probing i converges, that is, the limit $\lim_{h\to\infty} \frac{\sum_{t=1}^{h} \Pr[\sigma_t = i]}{h}$ exists. We limit our attention only to valid schedules.

2.2 Relating and optimizing objectives

The following lemma specifies the basic relation between the objectives. Its proof is straightforward.

Lemma 2.1. *For any schedule $\boldsymbol{\sigma}$,*

$$\textit{SUM}_e\text{:} \qquad \mathsf{E}_e\mathsf{M}_t[\boldsymbol{\sigma}] \geq \mathsf{M}_t\mathsf{E}_e[\boldsymbol{\sigma}] \geq \mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] \tag{2}$$

$$\textit{MAX}_e\text{:} \qquad \mathsf{M}_e\mathsf{M}_t[\boldsymbol{\sigma}] \geq \mathsf{E}_t\mathsf{M}_e[\boldsymbol{\sigma}] \geq \mathsf{M}_e\mathsf{E}_t[\boldsymbol{\sigma}] \tag{3}$$

For any objective we want to find schedules that minimize it. We denote the infimum of the objective over determinitic schedules by the prefix opt$_D$, over memoryless schedules by opt$_M$, and over stocastic schedules by opt. For example for the objective $\mathsf{M}_e\mathsf{E}_t$, opt$_D$-$\mathsf{M}_e\mathsf{E}_t$ is the infimum $\mathsf{M}_e\mathsf{E}_t$ over deterministic schedules. Since memoryless and determinitic schedules are a subset of stochastic schedules, the deterministic or the memoryless optima are always at least the stochastic optimum: For any objective opt$_D \geq$ opt and opt$_M \geq$ opt.

According to (2) and (3), the deterministic, memoryless, or stochastic optima of each objective satisfy the same relations. We show that for stochastic (and subsequently also for memoryless) schedules, all three objectives within each category (SUM$_e$ or MAX$_e$) are equivalent. We therefore use the streamlined notation opt-SUM$_e$ and opt-MAX$_e$ for the stochastic optima of all three SUM$_e$ or MAX$_e$ objectives and similarly opt$_M$-SUM$_e$ and opt$_M$-MAX$_e$ for the memoryless optima.

Lemma 2.2. .

$$\textit{opt-}\mathsf{E}_e\mathsf{E}_t = \textit{opt-}\mathsf{M}_t\mathsf{E}_e = \textit{opt-}\mathsf{E}_e\mathsf{M}_t \equiv \textit{opt-SUM}_e \tag{4}$$

$$\textit{opt-}\mathsf{M}_e\mathsf{E}_t = \textit{opt-}\mathsf{E}_t\mathsf{M}_e = \textit{opt-}\mathsf{M}_e\mathsf{M}_t \equiv \textit{opt-MAX}_e \tag{5}$$

Proof. We provide a high level proof sketch (full proof is in Appendix A): We take a stochastic schedule with certain $\mathsf{E}_e\mathsf{E}_t$ and "randomize" the start time by replacing it with a distribution of shifts of the schedule with "uniform" start time. The new stochastic schedule has for each element e, the same $T(e,t)$ for all times t which equals to $\mathsf{E}_t[e]$ of the original schedule. The technical difficulty handled in the full proof is that the schedule is infinite and we therefore need to work with a prefix. □

We show that optimizing any of our SUM$_e$ or MAX$_e$ objectives, over either stochastic or deterministic schedules is NP hard (proof in Appendix A)

Lemma 2.3. *Computing any one of the following optima is NP hard: opt$_D$-$\mathsf{E}_e\mathsf{E}_t$, opt$_D$-$\mathsf{M}_t\mathsf{E}_e$, opt$_D$-$\mathsf{E}_e\mathsf{M}_t$, opt-SUM$_e$, opt$_D$-$\mathsf{M}_e\mathsf{M}_t$, opt$_D$-$\mathsf{M}_e\mathsf{E}_t$, opt$_D$-$\mathsf{E}_t\mathsf{M}_e$, and opt-MAX$_e$.*

2.3 Deterministic versus stochastic

The distinction between objectives within each of the MAX_e and SUM_e groups only matters with deterministic scheduling. For an instance and objective, we attempt to understand the relation between the deterministic and stochastic optima. For deterministic MAX_e objectives, the compatison is to opt-MAX_e and for SUM_e objectives, it is to opt-SUM_e.

We show that on all instances, the deterministic $\mathsf{E}_e\mathsf{E}_t$ is equal to opt-SUM_e. Deterministic $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{M}_e\mathsf{M}_t$, however, are always strictly larger (proof in Appendix A).

Lemma 2.4.

$$opt_D\text{-}\mathsf{E}_e\mathsf{E}_t = opt\text{-}SUM_e \tag{6}$$
$$opt_D\text{-}\mathsf{E}_e\mathsf{M}_t \geq 2opt\text{-}SUM_e - 1 \tag{7}$$
$$opt_D\text{-}\mathsf{M}_e\mathsf{M}_t \geq 2opt\text{-}MAX_e - 1 \tag{8}$$

Additional relations, upper bounding the determinstic objective by the stochastic objective follow from relations with memoryless optima which are presented in the sequel.

3 Memoryless schedules

Memoryless schedules are particulary simple stochastic schedules specified by a probability distribution $\boldsymbol{q}$ on the tests. At each time, independently of history, we draw a test $i \in [m]$ at random according to $\boldsymbol{q}$ ($i \in [m]$ is selected with probability q_i) and probe i. It is easy to see that in memoryless schedules detection times are distributed geometrically. We show that memoryless schedules perform nearly as well, in terms of expected detection time, as general stochastic schedules. For notational convenience, we use the distribution $\boldsymbol{q}$ to denote also the memoryless schedule itself.

We first show that all SUM_e objectives and all MAX_e objectives are equivalent on any memoryless schedule.

Lemma 3.1. *For any memoryless schedule* $\boldsymbol{q}$*,*

$$\mathsf{E}_e\mathsf{M}_t[\boldsymbol{q}] = \mathsf{M}_t\mathsf{E}_e[\boldsymbol{q}] = \mathsf{E}_e\mathsf{E}_t[\boldsymbol{q}] = \sum_e \frac{p_e}{Q_e} \equiv SUM_e[\boldsymbol{q}]$$
$$\mathsf{M}_e\mathsf{M}_t[\boldsymbol{q}] = \mathsf{E}_t\mathsf{M}_e[\boldsymbol{q}] = \mathsf{M}_e\mathsf{E}_t[\boldsymbol{q}] = \max_e \frac{p_e}{Q_e} \equiv MAX_e[\boldsymbol{q}]\ ,$$

where $Q_e = \sum_{i|e\in s_i} q_i$.

Proof. The detection time of a fault on e via a memoryless schedule is a geometric random variable with paramter Q_e. In particular, for each element e, the distribution $T(e,t)$ are identical for all t and its expectation, $1/Q_e$, is equal to $\mathsf{M}_t[e]$ and $\mathsf{E}_t[e]$. From linearity of expectation, the $\mathsf{E}_e\mathsf{E}_t$, $\mathsf{M}_t\mathsf{E}_e$, and $\mathsf{E}_e\mathsf{M}_t$ are all equal to $\sum_e p_e/Q_e$. Similarly, $\mathsf{M}_e\mathsf{M}_t$, $\mathsf{M}_e\mathsf{E}_t$, and $\mathsf{E}_t\mathsf{M}_e$ are all equal to $\max_e \frac{p_e}{Q_e}$. $\square$

3.1 Memoryless Optima

We show that the memoryless optima with respect to both the SUM_e and MAX_e objectives can be efficiently computed. This is in contrast to deterministic and stochastic optima, which are NP hard.

Theorem 3.1. *The optimal memoryless schedule for SUM_e objectives, that is, the distribution* $\boldsymbol{q}$ *such that $SUM_e[\boldsymbol{q}] = opt_M\text{-}SUM_e$ is the solution of the convex program* (9) *(Figure 4).*

$$\text{minimize} \sum_e \frac{p_e}{\sum_{i|e\in s_i} q_i} \tag{9}$$
$$\forall i,\ q_i \geq 0$$
$$\sum_i q_i = 1$$

(a) Convex program for SUM$_e$

$$\text{maximize } z \tag{10}$$
$$\forall e,\ \frac{1}{p_e}\sum_{i|e\in s_i} q_i \geq z$$
$$\forall i,\ q_i \geq 0$$
$$\sum_i q_i = 1$$

(a) LP for MAX$_e$.

Figure 4: Computing SUM$_e$ and MAX$_e$ optimal memoryless schedules.

The optimal memoryless schedules with respect to the MAX$_e$objectives can be computed using an LP.

Theorem 3.2. *The optimal memoryless schedule for MAX$_e$, that is, the distribution **q** which satisfies MAX$_e[\boldsymbol{q}] =$ opt$_M$-MAX$_e$ is the solution of the LP* (10) *(Figure 4).*

Singletons instances: When each test is for a single element, the optimal solution of the convex program (9) has the frequencies of each element proportional to the square root of p_e [9], that is, $q_e = \sqrt{p_e}/\sum_e \sqrt{p_e}$. The SUM$_e$ optimum for an instance with weighting $\boldsymbol{p}$ is

$$\text{opt}_M\text{-SUM}_e(\boldsymbol{p}) = \sum_e \frac{p_e}{q_e} = \sum_e \sqrt{p_e} \sum_i \sqrt{p_i} = (\sum_i \sqrt{p_i})^2 \,. \tag{11}$$

In contrast, the solution of the LP (10) has optimal probing frequencies q_e proportional to p_e, that is, $q_e = p_e/\sum_e p_e$ and the MAX$_e$ optimum is opt$_M$-MAX$_e(\boldsymbol{p}) = \max_e \frac{p_e}{q_e} = \sum_e p_e$.

3.2 Memoryless versus Stochastic

For both SUM$_e$ and MAX$_e$ objectives, the optimum on memoryless schedules is within a factor of 2 of the optimum over general stochastic schedules.

Theorem 3.3.

$$\textit{opt-SUM}_e \leq \textit{opt}_M\textit{-SUM}_e \leq 2\textit{opt-SUM}_e \tag{12}$$
$$\textit{opt-MAX}_e \leq \textit{opt}_M\textit{-MAX}_e \leq 2\textit{opt-MAX}_e \tag{13}$$

Proof. The left hand side inequalities follow from memoryless schedules being a special case of stochastic schedules. To establish the right hand side inequalities, consider a stochastic schedule and let q_i be (the limit of) the relative frequency of test i. We have

$$\mathsf{M}_t[e] \geq \mathsf{E}_t[e] \geq \sum_e \frac{p_e}{2\sum_{i|e\in s_i} q_i}$$

Therefore, the average over elements $\sum_e \frac{p_e}{2\sum_{i|e\in s_i} q_i}$ must be at least half the optimum of (9) and the maximum $\max_e \frac{p_e}{2\sum_{i|e\in s_i} q_i}$ must be at least half the optimum of (10). $\square$

The following example shows that Theorem 3.3 is tight in that the "2" factors are realizable. That is, there are instances where the memoryless optimum is close to a factor of 2 larger than the respective stochastic optimum.

Lemma 3.2. *For any ϵ, there is an instance on which*

$$opt_M\text{-}MAX_e = opt_M\text{-}SUM_e \geq (2-\epsilon)opt\text{-}MAX_e = opt\text{-}SUM_e$$

Proof. The instance has n elements, corresponding n singleton tests, and uniform prioritiesp_e. The optimal memoryless schedule, the solution of both (10) and (9), has $q_e = 1/n$ and $\mathsf{M}_t[e] = \mathsf{E}_t[e] = n$ for each element. The optimal deterministic schedule repeats a permutation on the n elements and has $\mathsf{M}_t[e] = \mathsf{M}_e[t] = n$ and $\mathsf{E}_t[e] = \mathsf{E}_e[t] = (n+1)/2$ for all e, t. The optimal stochastic selects a permutation uniformly at random every n steps and follows it. It has $\mathsf{M}_t[e] = \mathsf{E}_t[e] = (n+1)/2$ for all elements. $\square$

3.3 Memoryless versus deterministic

Since a deterministic schedule is a special case of a stochastic schedule, from Theorem 3.3, the memoryless optimum is at most twice the deterministic optimum. The proof of Lemma 3.3 shows:

Lemma 3.3. *For any ϵ, there is an instance on which*

$$\begin{gathered} opt_M\text{-}MAX_e = opt_M\text{-}SUM_e = opt_D\text{-}\mathsf{M}_e\mathsf{M}_t = opt_D\text{-}\mathsf{E}_e\mathsf{M}_t = opt_D\text{-}\mathsf{E}_t\mathsf{M}_e \geq \\ (2-\epsilon)opt_D\text{-}\mathsf{M}_e\mathsf{E}_t = opt_D\text{-}\mathsf{E}_e\mathsf{E}_t = opt_D\text{-}\mathsf{M}_t\mathsf{E}_e = opt\text{-}MAX_e = opt\text{-}SUM_e \end{gathered}$$

That is, for the weaker SUM$_e$ and MAX$_e$ deterministic objectives, a gap of 2 is indeed realizable, meaning that it is possible for the deterministic optimum to be smaller than the respective memoryless optimum. For the strongest objectives, $\mathsf{E}_e\mathsf{M}_t$ for SUM$_e$ and $\mathsf{M}_e\mathsf{M}_t$ for MAX$_e$, we show that the deterministic optimum is at least the memoryless optimum:

Lemma 3.4.

$$\begin{gathered} opt_M\text{-}SUM_e \leq opt_D\text{-}\mathsf{E}_e\mathsf{M}_t \\ opt_M\text{-}MAX_e \leq opt_D\text{-}\mathsf{M}_e\mathsf{M}_t \end{gathered}$$

Proof. Similar to the proof of Theorem 3.3: Consider a deterministic schedule and let q_i be (the limit of) the relative frequency of test i. We have $\mathsf{M}_t[e] \geq \frac{p_e}{\sum_{i|e\in s_i} q_i}$. $\square$

The other direction, upper bounding the deterministic optimum by the memoryless optimum, is extensively dealt with in the following sections.

4 Deterministic scheduling

For a deterministic schedule and an objective, the *approximation ratio*, which for consistency we also refer to as the D2D, is the ratio of the objective on the schedule to that of the (determinsitic) optimum of the same objective. We are ultimately interested in efficient constructions of deterministic schedules with good approximation ratio and in quantifying the cost of determinism, that is, asking how much worse a deterministic objective can be over the stochastic objective. The *D2S ratio* (or D2S in short) of a deterministic schedule as the ratio of the objective on the schedule to that of the *stochastic* optimum of the same objective.

Since both deterministic and stochastic optima are NP hard to compute, so is the approximation ratio and the D2S. We therefore relate the performance of a deterministic schedule to the memoryless optimum, using the D2M, which we define as the ratio of the objective on the schedule to that of the *memoryless* optimum of the same objective. The D2M of any given schedule can be computed efficiently by computing the memoryless optimum. Moreover, it can be used to bound the hard to compute D2D and D2S: The relation D2S $\leq$ D2M ≤ 2 D2S follows from Theorem 3.3. The D2D is upper bounded by D2D ≤ 2 D2M (For $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{M}_e\mathsf{M}_t$, we have D2D $\leq$ D2M see Lemma 3.4). Accordingly the D2M $\geq 1/2$ and for $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{M}_e\mathsf{M}_t$ we have D2M ≥ 1. The optimum D2M is the minimum possible over all schedules. We refer to the supremum of optimum D2M over instances as the *D2M gap* of the scheduling problem.

Our proposed constructions of the R-Tree and Kuhn-Tucker deterministic schedulers are presented in detail in the following sections. Our analytic results are summarized in Table 1. The lower bounds on the D2M gap are established in Lemma 4.2 through example instance on which the optimum D2M is large. The lower bound on approximability is established in Lemma 4.3. Both lower and upper bounds for $\mathsf{E}_e\mathsf{E}_t$ and $\mathsf{M}_e\mathsf{E}_t$ are established in Lemma 4.1, $\mathsf{E}_e\mathsf{M}_t$ D2M upper bound in Theorem 5.1, and $\mathsf{M}_e\mathsf{M}_t$ D2M in Theorem 5.2.

objective	scheduling D2M	D2M gap	approximability
$\mathsf{E}_e\mathsf{E}_t$	1	1	
$\mathsf{M}_e\mathsf{E}_t$	1	1	
$\mathsf{E}_e\mathsf{M}_t$	$O(\ln m)$	$\Omega(\ln m)$	
$\mathsf{M}_e\mathsf{M}_t$, $\mathsf{E}_t\mathsf{M}_e$	$O(\log n + \log \ell)$	$\Omega(\log n)$, $\Omega(m)$	$\Omega(\log n)$

Table 1: The D2M guaranteed by our constructions (in particular, this implies upper bound on the D2M), and lower bounds on the D2M gap and on efficient approximability.

4.1 OPT$_D$-$\mathsf{E}_e\mathsf{E}_t$ and OPT$_D$-$\mathsf{M}_e\mathsf{E}_t$

For the objectives $\mathsf{E}_e\mathsf{E}_t$ and $\mathsf{M}_e\mathsf{E}_t$, which are respectively the weakest SUM$_e$ and MAX$_e$ objectives, we show that the deterministic optimum is at most the memoryless optimum. Moreover, we can efficiently construct deterministic schedules with D2M arbitrarily close to 1 (and thus approximation ratio of at most 2).

Lemma 4.1.

$$\textit{opt}_D\text{-}\mathsf{M}_e\mathsf{E}_t \leq \textit{opt}_M\text{-}\textit{MAX}_e$$
$$\textit{opt}_D\text{-}\mathsf{E}_e\mathsf{E}_t \leq \textit{opt}_M\text{-}\textit{SUM}_e$$

and for any $\delta > 0$ we can efficiently construct deterministic schedules with $\mathsf{M}_e\mathsf{E}_t$ or $\mathsf{E}_e\mathsf{E}_t$ D2M $\leq (1+\delta)$.

Proof. For any ϵ, for a long enough run of the memoryless schedule $\boldsymbol{q}$, there is a positive probability that for all elements, the average $T(e,t)$ is within $(1+\epsilon)$ of its expectation $\mathsf{E}_t[e|\boldsymbol{q}]$. When the schedule is long with respect to $\max_e \mathsf{E}_t[e|\boldsymbol{q}]$ we can obtain a deterministic schedule which "cycles" through it while that property. The schedule has $\mathsf{M}_e\mathsf{E}_t \leq (1+\epsilon)\text{opt}_M\text{-MAX}_e$. □

4.2 Lower bounds on D2M gap

In contrast, for the strongest objectives, $\mathsf{M}_e\mathsf{M}_t$, $\mathsf{M}_e\mathsf{E}_t$, and $\mathsf{E}_e\mathsf{M}_t$, we construct a family of instances with asymptotically large optimal D2M.

Lemma 4.2. *There is a family of instances with m tests and n elements such that each element participates in ℓ tests with the following lower bounds on D2M: The $\mathsf{E}_t\mathsf{M}_e$-D2M (and thus $\mathsf{M}_e\mathsf{M}_t$-D2M) $\Omega(\ln n)$ and*

$\Omega(m)$. *The* $\mathsf{E}_e\mathsf{M}_t$ *D2M is* $\Omega(\log \ell)$. *Moreover, these instances can be realized on a network, where elements are links and tests are paths.*

4.3 Approximability lower bounds for opt$_D$-$\mathsf{M}_e\mathsf{M}_t$ and opt$_D$-$\mathsf{E}_t\mathsf{M}_e$

Lemma 4.3. *The problems opt*$_D$-$\mathsf{M}_e\mathsf{M}_t$ *and opt*$_D$-$\mathsf{E}_t\mathsf{M}_e$ *are hard to approximate to anything better than* $\ln(n)$.

Proof. When $\boldsymbol{p}$ is uniform, opt$_D$-$\mathsf{M}_e\mathsf{M}_t$ is equivalent to set cover – an approximation ratio for opt$_D$-$\mathsf{M}_e\mathsf{M}_t$ implies the same approximation ratio for set cover [11], which is hard to approximate [6].

This also extends to opt$_D$-$\mathsf{E}_t\mathsf{M}_e$, again using uniform $\boldsymbol{p}$. A minimum set cover of size k implies a schedule (cycling through the cover) with $\mathsf{E}_t\mathsf{M}_e$ of k. Also, a schedule with $\mathsf{E}_t\mathsf{M}_e$ at most k means that $\mathsf{M}_e[t] \leq k$ for at least one t, means there is a cover of size k. □

5 R-Tree schedules

We present an efficient construction of deterministic schedules from a distribution $\boldsymbol{q}$ and relate detection times of the deterministic schedule to (expected) detection times of the memoryless schedule defined by $\boldsymbol{q}$.

We can tune the schedule to either MAX$_e$ or SUM$_e$ objectives, by selecting accordingly the input frequencies $\boldsymbol{q}$ as a solution of (10) or (9). We then derive analytic bounds on the D2M of the schedules we obtain.

The building block of *random tree* (R-Tree) schedules is *tree schedules*, which are deterministic schedules specified by a mapping of tests to nodes of a binary tree. A tree schedule is specified with respect to probing frequencies $\boldsymbol{q}$ and has the property that for any test, the maximum probing interval in the deterministic schedule is guaranteed to be close to $1/q_i$. However, if we do not place the tests in the tree carefully then for an element covered by multiple tests the probing interval can be close to that of its most frequent test, but yet far from the desired (inverse of) $Q_e = \sum_{i|e\in s_i} q_i$. Therefore, even when computed with respect to $\boldsymbol{q}$ which solves (9), the tree schedule can have $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{E}_e\mathsf{E}_t$ D2M ratios $\Omega(\ell)$.

We define a distribution over tree schedules obtained by randomizing the mapping of tests to nodes. We then bound the expectation of the $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{E}_e\mathsf{E}_t$ (when applied to $\boldsymbol{q}$ which solves (9)) and $\mathsf{M}_e\mathsf{M}_t$ (when applied to $\boldsymbol{q}$ which solves (10)) over the resulting deterministic schedules. Given a bound on the expectation of an objective, there is a constant probability that a tree schedule randomly drawn from the distribution will satisfy the same bound (up tp a small constant factor). An R-Tree schedule is obtained by constructing multiple tree schedules drawn from the distribution, computing the objectives on these schedules, and finally, returning the best performing tree schedule. Note that even though the construction is randomized, the end result, the R-Tree schedule, is deterministic, since it is simply a tree schedule.

Specifically, lets take SUM$_e$as an example, we apply the R-Tree schedule construction several times with $\boldsymbol{q}$'s solving (9). The tree with the best $\mathsf{E}_e\mathsf{M}_t$ has $O(\log(\ell))$ $\mathsf{E}_e\mathsf{M}_t$ D2M and the tree with the best $\mathsf{E}_e\mathsf{E}_t$ has a constant $\mathsf{E}_e\mathsf{E}_t$ D2M. Furthermore we can also find a tree which satisfies both guarantees.

Theorem 5.1. *A deterministic schedule with* $\mathsf{E}_e\mathsf{M}_t$ *D2M ratio of* $O(\log \ell)$ *and a constant* $\mathsf{E}_e\mathsf{E}_t$ *D2M ratio can be constructed efficiently.*

The theorem is tight since from Lemma 4.2, the $\mathsf{E}_e\mathsf{M}_t$ D2M gap on some instances is $\Omega(\log \ell)$, and therefore, we can not hope for a better dependence on ℓ.[3]

For MAX$_e$, we show that when we apply the R-Tree schedule construction to $\boldsymbol{q}$ which is the optimum of (10), we obtain a deteministic schedule with $O(\log \ell + \log n)$ $\mathsf{M}_e\mathsf{M}_t$ D2M.

[3]As a side note, recall that according to (6) there exist schedules with $\mathsf{E}_e\mathsf{E}_t$ D2M close to 1, so with respect to $\mathsf{E}_e\mathsf{E}_t$ this only shows that we can simultaneously obtain a $\mathsf{E}_e\mathsf{M}_t$ D2M that is logarithmic in ℓ and at the same time a constant $\mathsf{E}_e\mathsf{E}_t$ D2M.

Theorem 5.2. *A deterministic schedule with $\mathsf{M}_e\mathsf{M}_t$ D2M ratio of $O(\log \ell + \log n)$ can be constructed efficiently.*

From Theorems 5.1 and 5.2, we obtain the following upper bounds on the D2M gap and efficiently construct deterministic schedules satisfying these bounds (summarized in Table 1).

$$\begin{aligned} \text{opt}_D\text{-}\mathsf{E}_e\mathsf{M}_t &= O(\log \ell)\text{opt-SUM}_e \\ \text{opt}_D\text{-}\mathsf{M}_e\mathsf{M}_t &= O(\log \ell + \log n)\text{opt-}\mathsf{M}_e\mathsf{M}_t \end{aligned}$$

5.1 Tree schedules

A tree schedule is a deterministic schedule guided with frequencies $\boldsymbol{q}$ where probes to test i are spaced $[1/q_i, 2/q_i)$ probes apart. When q_i has the form $q_i = 2^{-j}$, test i is performed regularly with period 2^j.

Assume for now that $q_i = 2^{-L_i}$ for positive integer L_i for all i. We map each i to nodes of a binary tree where i is mapped to a node at level L_i and no test can be a child of another. This can be achieved by greedily mapping tests by decreasing level – we greedily map tests with level $L_i = 1$, then tests with $L_i = 2$ and so on. Once a test is mapped to a node, its subtree is truncated and it becomes a leaf.

From this mapping, we can generate a deterministic schedule as follows: The sequence is built on alternations between left and right child at each node. Each node "remembers" the last direction to a child. To select a test, we do as follows. First visit the root and select the child that was not visited previous time. If a leaf, we are done, otherwise, we recursively select the child that was not previously visited and continue. This until we get to a leaf. We then output test i. This process changed "last visit" states on all nodes in the path from the root to the leaf. It is easy to see that if a leaf at level L is visited once every 2^L probes. An example of a set of frequencies, a corresponding mapping, and the resulting schedule is provided in Figure 5.

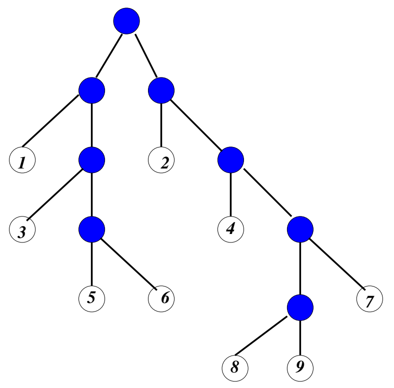
If probabilities are of general form, we can map each test according to the highest order significant bit (and arbitrarily fill up the tree). When doing this we get per-test ratio between the actual and desired probing frequencies of at most 2. Alternatively, we can look at the bit representation of q_i– separately map all "1" positions in the first few significant bits to tree nodes. In this case the average probing frequency of each test is very close to q_i but the maximum time between probes depends on the relation between the tree nodes to which the bits of test i are mapped to. The only guarantee we have on the maximum is according to the most significant bit $2^{-\lceil \log_2(1/q_i) \rceil}$. Under "random" mappings the expectation of the maximum gets closer to the average.

Singleton tests. For a given instance, the best D2M we can hope for is when the deterministic scheduler is able to perform each test in precise intervals of $1/q_i$, which results, for singletons instances, in maximum probing interval of $1/q_i$. Tree schedules achieve this when $q_i = 2^{-L_i}$ for all i. A deterministic tree schedule for singletons has D2M that is at most 2, and therefore, for all our objectives, the D2M gap is at most 2.

The $\mathsf{M}_e\mathsf{M}_t$ D2M gap and the $\mathsf{E}_e\mathsf{M}_t$ D2M gap, however, are *exactly* 2. Consider an instance with two elements one with priority $p_1 = 1 - \epsilon$ and the other with priority $p_2 = \epsilon$.

Consider $\mathsf{M}_e\mathsf{M}_t$. The optimal memoryless schedule (10) has $q_1 = 1 - \epsilon$ and $q_2 = \epsilon$ and $\max_e p_e\mathsf{M}_t[e] = 1$. Whenever there are at least two elements with positive priorities, any deterministic scheduler has $\mathsf{M}_t[e] \geq 2$ for all elements. Therefore, the $\mathsf{M}_e\mathsf{M}_t$ of any deterministic schedule is at least 2 and the D2M is at least 2.

Consider $\mathsf{E}_e\mathsf{M}_t$. The optimal memoryless schedule (9) has $q_1 = \frac{\sqrt{1-\epsilon}}{\sqrt{1-\epsilon}+\sqrt{\epsilon}}$ and $q_2 = \frac{\sqrt{\epsilon}}{\sqrt{1-\epsilon}+\sqrt{\epsilon}}$ and the $\mathsf{E}_e\mathsf{M}_t = p_1/q_1 + p_2/q_2 = (\sqrt{1-\epsilon} + \sqrt{\epsilon})^2 \approx 1$. A deterministic schedule has $\mathsf{M}_t[e] \geq 2$ for both elements and thus $\mathsf{E}_e\mathsf{M}_t = p_1\mathsf{M}_t[1] + p_2\mathsf{M}_t[2] = 2$. It folows that the $\mathsf{E}_e\mathsf{M}_t$ D2M ration is $\geq 2 - \epsilon$ for any small $\epsilon > 0$.



$q_1 = q_2 = 1/4$, $q_3 = q_4 = 1/8$, $q_5 = q_6 = q_7 = 1/16$, $q_8 = q_9 = 1/32$

L	schedule															
2	1	2	x	x												
3	1	2	3	4	1	2	x	x								
4	1	2	3	4	1	2	5	x	1	2	3	4	1	2	6	7
5	1	2	3	4	1	2	5	8	1	2	3	4	1	2	6	7
-	1	2	3	4	1	2	5	9	1	2	3	4	1	2	6	7

Figure 5: Mapping tests to nodes of a binary tree to produce a deterministic schedule. The table shows the level-L schedule for $L = 2, 3, 4, 5$. The full deterministic schedule cycles through the level-5 schedule.

Several deterministic schedules for singletons with ratio at most 2 (and better than 2 when possible for the particular instance, in particular when priorities are small) were previously proposed [3, 4]. Tree schedules are of interest to us here because they can be "properly" randomized to yield good performance in our treatment of general instances.

5.2 Random tree schedules

Consider an instance and a memoryless schedule with frequencies $\boldsymbol{q}$. We assume that q_i have the form 2^{-L_i} for positive integers L_i (this is without loss of generality as we can only look at the highest order bit and loose a factor of at most 2). We construct a tree schedule for $\boldsymbol{q}$ by mapping the tests to nodes randomly as follows. We process tests by increasing level. In each step (level), all tests of the current level are randomly mapped to the available tree nodes at that level. After a test is mapped to a node, its subtree is truncated.

For each level N (which can be at most the maximum L_i), we can consider the *level-N schedule*, which is a cyclic schedule of length 2^N. The schedule specifies the probes for all tests with level $L_i \leq N$, and leaves some spots "unspecified".

We now specify the level-N schedule of the tree. Consider a completion of the tree to a full binary one with 2^N leaves (truncate everything below level N). Associate with each leaf a a binary number $\overline{a}$ which contains a 0 at digit i (from right to left, i.e. the least significant digit correspond to the child of the root and the most significant digit corresponds to the leaf itself) if the ith child on its path from the root is a left child. We refer to $\overline{a}$ as the *position* of leaf a.

We construct the sequence by associating test i with all leaf descendants of the node containing it, and with all the positions of the sequence corresponding to these leaves. Putting it in another words the level-N schedule of the tree cycles through the leaves a at level-N (of the completion of the tree) according to the order defined by $\overline{a}$ and probes the test associated with each leaf. A test with $q_i = 2^{-L_i}$ is probed in regular intervals of 2^{L_i}. The first probe is distributed uniformly at random from $[0, 2^{L_i} - 1]$.

Level-N schedules constructed from the same mapping for different depths N are consistent in the following sense: The level $N' > N$ schedule is $2^{N'-N}$ repetitions of the level-N schedule in terms of the tests specified by a level-N schedule (those with level $L_i \leq N$) and also specifies tests with $N < L_i \leq N'$.

5.3 Analysis of R-Tree schedules

For a single element e, we analyze the expected (over our randomized construction of a deterministic tree schedule) maximum probe interval in the deterministic schedule. We show

Lemma 5.1. *The expected maximum is $\Theta(\log \ell_e)/Q_e$, where $\ell_e = \{i \mid e \in s_i\}$. I.e., for any element e, $E_{algo}[\max_t T(e,t)] \leq c\log(\ell_e)/Q_e$, where $T(e,t)$ is the elapsed time from time t until e is probed.*

Proof. Given a level N schedule, we say that a subinterval of $[0, 2^N - 1]$ is *hit by a test* if contains a leaf of the test. We say it is hit by an element e if it is hit by at least one test containing the element.

Consider a particular element e. We now look only at the tests which include the element. To simplify notation, let q_i, $i \in [\ell_e]$ be the frequencies of these tests, let $Q = \sum q_i$, and $q_{\max} = \max_i q_i$.

We consider the schedule for some level $N \in [\log_2(1/q_{max}), \max_i L_i]$. We will make a precise choice of N later on.

Note that any interval of size $\geq 1/q_{\max}$ must be hit by the test with maximum frequency. We are now looking to bound the distribution of the size of the largest interval that is not hit.

Consider now a subinterval $\subset [0, 2^N - 1]$ of size $D < 1/q_{\max}$. We can assume that $D = 2^j$ for some j and the interval left endpoint is an integral multiple of D.

We upper bound the probability that the interval it is not hit by e. The probability that it is not "hit" by a test with frequency q_i is q_iD. These probabilities of not hitting the interval by different tests are negatively correlated: conditioned on some of the tests not hitting the interval, it only makes it more likely that other tests do hit the interval – hence, the probability that the interval is not hit by any test is at most the product $\prod_i(1 - q_iD)$, which in turn is bounded from above by $\prod_i(1 - q_iD) \leq \exp(-\sum q_iD) = \exp(-QD)$.

We now upper bound the probability that there exists at least one subinterval of size $D = 2^j$ and left endpoint that is an integral multiple of D, that is not hit by any test. We do a union bound on $2^N/D$ intervals of this property and this probability is at most

$$\frac{2^N}{D}\exp(-QD) \; . \tag{14}$$

Note that if using $D = \frac{x}{2}$, this upper bounds the probability that there exists an interval of size x that is not hit (without restrictions on endpoints). This probability, in terms of x, is

$$\frac{2^{N+1}}{x}\exp(-Qx/2) \tag{15}$$

We now restrict our attention to a subset S of the tests which satisfy $q_i \geq \frac{Q}{2\ell_e}$. We have $Q_S \equiv \sum_{i \in S} q_i \geq Q/2$. We now look only at the tests in S. since this is a subset of the tests that include e, it is sufficient to bound the expectation of the largest open interval with respect to these tests. Since the highest level in S is $N = \lceil \log_2(2\ell_e/Q) \rceil \leq 1 + \log_2(\ell_e/Q)$, we can look at the level N schedule. We substitute this N and $Q_S \geq Q/2$ in (15) we obtain that the probability of an empty interval of size x is

$$\frac{8\ell_e}{xQ}\exp(-xQ/4) \; . \tag{16}$$

For $x = 8\ln \ell_e/Q$ in (16), we obtain a bound of $1/(\ell_e \ln \ell_e) \leq 1/2$ (for $\ell_e \geq 2$, $\ell_e = 1$ is already covered as q_{max}).

We can now obtain an upper bound on the expectation of the maximum empty interval by summing over positive integers i, the product of interval size $(i+1)x$ and an upper bound on the probability of an empty interval of at least size ix, for positive integer i, we obtain that the expectation is $O(x) = (1/Q)O(\ln \ell_e)$. $\square$

Proof of Theorem 5.1

Proof. We start with frequencies $\boldsymbol{q}$ and build a deterministic tree schdule using our randomized construction. We show that the expected $\mathsf{E}_e\mathsf{M}_t$ of the deterministic schedule that we obtain is at most $\Theta(\ln \ell)$ times the $\mathsf{E}_e\mathsf{M}_t$ of the memoryless schedule for $\boldsymbol{q}$. To obtain our claim, we take $\boldsymbol{q}$ to be the optimum of (9).

We apply Lemma 5.1. The lemma shows that for each element e we have $E_{alg}[\max_t T(e,t)] \leq c\log(\ell_e)/Q_e$. Now we take a weighted sum over elements using $\boldsymbol{p}$. We get that,

$$E_{e\sim p_e}E_{alg}[\max_t T(e,t)] \leq \sum_e p_e \frac{c\log(\ell_e)}{Q_e}$$

This is equivalent to,

$$E_{alg}E_{e\sim p_e}[\max_t T(e,t)] \leq \sum_e c\log(\ell_e)\frac{p_e}{Q_e} \leq c\log(\ell_{\max})\sum_e \frac{p_e}{Q_e}$$

This implies that with probability at least $1/2$ (over the coin flips of the algorithm) we get a deterministic schedule whose $\mathsf{E}_e\mathsf{M}_t$ is $2c\log(\ell_{\max})\sum_e \frac{p_e}{Q_e}$. It follows that the $\mathsf{E}_e\mathsf{M}_t$ D2M ratio is at most $2c\log(\ell_{\max})$.

We now show that the $\mathsf{E}_e\mathsf{E}_t$ D2M ratio of a random tree schedule is constant with constant probability. Using the same reasoning as in the proof above for $\mathsf{E}_e\mathsf{M}_t$ it suffices to show that for each e, $E_{alg}E_t[T(e,t)] \leq c/Q_e$.

Fixing e and an arbitrary time t, as in the proof of Lemma 5.1, we can easily derive that $\Pr[T(e,t) \geq D] \leq \exp -Q_eD$. In particular we get that $\Pr[T(e,t) \geq i/Q_e] \leq \exp(-i)$. So the fraction of times t in which $T(e,t) \geq i/Q_e$ is at most $\exp(-i)$. It follows that $E_{alg}E_t[T(e,t)] \leq (2/Q_e)\sum_i \exp(-i) \leq c/Q_e$ for some constant c. $\square$

Proof of Theorem 5.2

Proof. We use (16) in the proof of Lemma 5.1. For an element e, the probability of an empty interval of size at least x is at most $\frac{8\ell_e}{xQ}\exp(-xQ/4)$. Using $x \equiv D_e = 8(\ln n + \ln \ell_e)/Q$ we obtain that there is an interval empty of tests for e of length at least D_e with probability at most $1/n^2$.

By the probability union bound over the elements we get that the probability that for all e there is no empty interval of length more than D_e is at least $1 - 1/n$. $\square$

6 Kuhn-Tucker inspired schedule

The Kuhn-Tucker conditions on the optimal solution of our convex program (9) translate to

$$r_i = \frac{\partial \sum_e \frac{p_e}{\sum_{i|e\in s_i} q_i}}{\partial q_i} = -\sum_{e|e\in s_i} \frac{p_e}{(\sum_{i|e\in s_i} q_i)^2} .$$

being balanced.

We suggest a deterministic greedy heuristic for SUM$_e$, illustrated in Algorithm 1, which is based on that. For each element e, we track x_e which is the elapsed number of probes since e was last probed. We then choose the test i with maximum $\sum_{e|e\in s_i} p_e(x_e+1)^2$.

We conjecture that the KT schedule has $\mathsf{E}_e\mathsf{E}_t$ which is at most twice the optimal. Viewing the quantitiy $\sum_e p_e(x_e+1)^2$ as "potential" the average reduction in potential is the $\mathsf{E}_e\mathsf{E}_t$ of the sequence. We do not provide bounds on the approximation ratio, but test this heuristic in our experiments.

Algorithm 1 Kuhn-Tucker (KT) schedule

```
 1: function BEST-TEST
 2:     $v \leftarrow 0$
 3:     for $s \in \mathcal{S}$ do
 4:         $y \leftarrow 0$
 5:         for $e \in s$ do
 6:             $y \leftarrow y + x[e]$
 7:         if $(y > v)$ then
 8:             $b \leftarrow s$; $v \leftarrow y$
        return $b$                                          ▷ best test

 9: function KT-SCHEDULE($V, p, \mathcal{S}$)
10:     for $e \in V$ do
11:         $x[e] \leftarrow 0$
12:     while True do
13:         $s \leftarrow$BEST-TEST()
14:         output $s$
15:         for $e \in s$ do
16:             $x[e] \leftarrow 0$
```

The KT scheduler can be deployed when priorities are modified. This is in contrast to other schedulers which pre-compute the schedule .

7 Extension to probabilistic tests

A useful extension of our model allows for a probability π_{ei} that depends on i and e that a failure to e is found with test i. We assume that different probes invoking the same or different tests are independent. Probabilistic tests can model ECMP (equal cost multi-paths) and transient (inconsistent) failures: Transient failures are modeled by a fixed probability $\pi_{ei} \in (0,1]$ of packet loss. Tests under ECMP are modeled by s_i being a unit flow between the origin and destination that defines a probability distribution over tests, where the "flow" traversing e is π_{ei}.

With probabilistic tests, we may as well use stochastic schedules, in particular, memoryless schedules, which also offer strong guarantees on the variance of detection times. Our models and results for memoryless schedules have straightforward extensions to probabilistic tests. The convex program for opt$_M$-SUM$_e$ can be modified to incorporate probabilistic tests if we replace in (9) $\sum_{i|e\in s_i} q_i$ by $\sum_i \pi_{ei}q_i$. The LP for opt$_M$-MAX$_e$ can be modified by replacing in (10) for each element e $\sum_{i|e\in s_i} q_i$ by $\sum_i \pi_{ei}q_i$.

SUM_e in memoryless schedulers:

algorithm	GN-U	GN-P	GN-Z	Clos
Convex	95.66	59.72	25.92	32.02
LP	105.29	68.77	118.38	32.02
Uniform	229.16	72.27	260.46	33.00
SAMP SC	111.56	82.70	86.17	32.00
SAMP KT	108.54	61.45	86.17	32.00

$\mathsf{E}_e\mathsf{E}_t$ in deterministic schedulers:

algorithm	GN-U	GN-P	GN-Z	Clos
SC	60.27	49.42	51.52	16.50
KT	58.04	33.93	14.63	16.50
RT CON	66.43	49.21	17.92	30.76
RT LP	85.47	63.81	88.07	31.00
RT-S CON	57.87	46.91	18.69	
RT-S LP	59.70	50.24	87.47	

$\mathsf{M}_t\mathsf{E}_e$ in deterministic schedulers:

algorithm	GN-U	GN-P	GN-Z	Clos
SC	70.43	62.08	93.80	16.50
KT	70.04	62.08	20.36	16.50
RT CON	72.23	56.53	24.65	36.05
RT LP	95.81	73.34	113.47	36.20
RT-S CON	60.08	50.02	23.38	
RT-S LP	63.09	53.82	96.67	

$\mathsf{E}_e\mathsf{M}_t$ in deterministic schedulers:

algorithm	GN-U	GN-P	GN-Z	Clos
SC	124.93	109.46	114.29	32.00
KT	130.11	93.02	35.34	32.00
RT CON	180.00	179.91	53.40	144.14
RT LP	319.12	261.65	269.01	146.70
RT-S CON	121.24	103.91	42.61	
RT-S LP	123.35	107.42	183.89	

Table 2: SUM_e objectives. Table shows expected time with memoryless schedules (same for all SUM_e objectives) and $\mathsf{E}_e\mathsf{E}_t \leq \mathsf{M}_t\mathsf{E}_e \leq \mathsf{E}_e\mathsf{M}_t$ on different deterministic schedulers.

8 Experimental Evaluation

We evaluated the performance of our schedulers for testing for silent link failures in two networks. The first is a backbone network (denoted GN in the sequel) of a large enterprise. We tested 500 of the network links with 3000 MPLS paths going through them.

The second network we considered is a (very regular) folded Clos network (denoted Clos) of 3 levels and 2048 links. On this network we considered all paths between endpoints. The Clos network is a typical interconnection network in data centers.

For the Clos network, we only considered uniform weights (priorities), meaning that all links are equally important. For the GN network, we considered uniform weights (denoted GN-U), weights that are proportional to the number of MPLS paths traversing the link (GN-P, where P designates popularity), and Zipf distributed weights with parameter 1.5 (GN-Z).

On these four networks (links and paths with associated weights), Clos, GN-U, GN-P, and GN-Z, we simulated our schedulers and evaluated their performance with respect to the different objectives.

Memoryless schedulers: We solved the convex program (9) for SUM_e objectives and the LP (9) for MAX_e objectives to obtain optimal memoryless probing frequencies $\boldsymbol{q}$. These optimization problems were solved using Matlab and CVX [4].

We compared these optimal memoryless schedules to other memoryless schedules obtained using three naive selections of probing frequencies: the first is uniform probing of all paths (Uniform), the second is uniform probing of a smaller set of paths that cover all the links (SAMP SC), and the third is probing according to frequencies generated by the Kuhn-Tucker schedule (SAMP KT).

[4] See http://cvxr.com/cvx/.

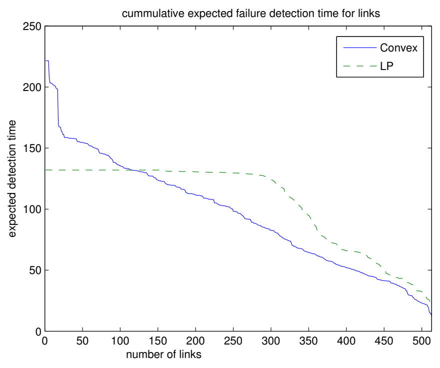


Figure 6: Distribution of time to detect a fault of a link in the backbone network with equal weights.

The performance of these schedules, in terms of the expected detection times $T(e,t)$ is shown in Table 2 (SUM$_e$ objective) and Table 3 (MAX$_e$ objective). The schedulers optimized for one of the objectives, SUM$_e$ or MAX$_e$, clearly dominate all others with respect to the objective it optimizes. We can see that while on some instances the alternative schedulers are close to the optimal one, on others performance gaps are substantial. In particular, even with respect to the scheduler optimizing the other objective. This shows the importance of selecting appropriate objective and optimizing for it.

To better understand the difference between the SUM$_e$ and MAX$_e$ objectives, we depicted in Figure 6 a reverse CDF of the average expected time to detect a failure in a link of the backbone network with uniform weights (GN-U). One can see that the LP, which optimizes the worst-off link, has a smaller maximum value but the average, which corresponds to the area below the curve, is larger than for the convex program, which optimizes the average link.

Deterministic schedulers: Next we evaluated the performance of the deterministic schedulers. Here, $T(e,t)$, the wait from time t till the next path containing e is scheduled, is deterministic. We used two different implementation of the R-Tree algorithm (Section 5). In the first, the algorithm was seeded with the frequencies computed by the LP (RT LP) or by the convex program (RT CON) when applied to the full set of paths. We discuss the second implementation in the sequel. We also implemented the Kuhn-Tucker (KT) scheduler (Section 6), and a simple greedy Set Cover algorithm (SC) which was previously proposed for the $\mathsf{M}_e\mathsf{M}_t$ metric (minimum set cover is the optimal deterministic scheduler for $\mathsf{M}_e\mathsf{M}_t$ when priorities are uniform). This scheduler produces a cyclic sequence consisting of a set cover produced by the greedy approximation algorithm for set cover.

Table 2 shows the values of all SUM$_e$ objectives for the different memoryless and deterministic schedulers and Table 3 shows the same for the MAX$_e$ objectives. One can easily see that indeed all results follow the ratios between the different objectives (for example $M_tE_e \geq E_eE_t$ or $M_eE_t \leq E_tM_e$). Moreover, while the SC algorithm works well for equal weights it performs fairly bad across all objectives for GN-Z.

When priorities are uniform (as in Clos and GN-U), minimum set cover produces the optimal deterministic schedule for $\mathsf{M}_e\mathsf{M}_t$ and $\mathsf{E}_t\mathsf{M}_e$. We can see that indeed SC performs very well. When priorities are highly skewed, however, its performance deteriorates.

The KT scheduler performed well on the SUM$_e$ objectives, which it is designed for. Because its adaptive nature, rather than precomputing a fixed schedule, it is recommended for applications where priorities are changing on the go, such as when the priority of an element correspond to current traffic levels traversing the element.

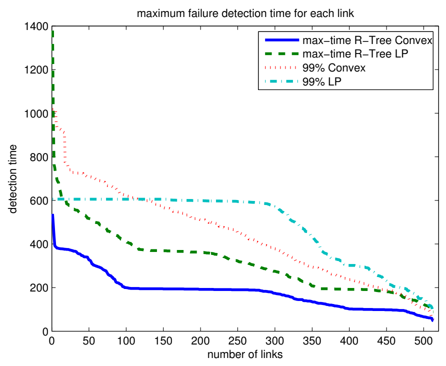


Figure 7: Distribution of time to detect a fault: RT LP and RT CON (deterministic) vs. memoryless over GN-Us.

Our R-Tree schedulers (RT CON and RT LP) did not perform well on some of the instances, and in some cases, performed worse than SC and KT. The reason, as the analysis shows, is the logarithmic dependence on ℓ, which in our case, is the number of tests used to cover the link in the solution of the LP and convex programs. The original collection of paths turned out to have high redundancy, where subpaths have many alternatives and the fractional solvers tend to equally use all applicable paths. We can see evidence for this fragmentation in Figure 6.

To address this issue, we seeded the R-Tree algorithm with respective solutions of the LP and Convex programs applied to a modified instance with a pre-selected small subset of the original paths. The subset was picked so that it contains a cover of the links and also tested to ensure that the objective of the optimization problem does not significantly increase when implementing this restriction. On those instances, tests which constitute a set cover of the links and produced by the greedy approximation algorithm, performed well. We denote the respective schedulers obtained this way using the LP and convex solutions, by RT-S LP and RT-S CON.

The results of this experiment are included in Tables 2 and 3. We can observe that this heuristic improves the performance of the R-Tree algorithm substantially for all objectives. The value of SUM_e (for the memoryless schedule produce by solving the convex program on this subset) has increased from 95.66 to 104.33 and the value of MAX_e had increased from 132.05 to 142.01. We leave the question of how to choose the subset to best balance the loss in the objective of the memoryless schedule with the gain in better derandomization for further research.

Memoryless vs. Deterministic: In many practical scenarios, memoryless schedulers have considerable advantage, due to their statelessness. However, due to their stochastic nature, we only have guarantees on the expectation. Deterministic schedulers, on the other hand, require a centralized implementation but can provide worst case guarantees regarding the time (or weighted cost) until detecting a failure. We demonstrate this issue by illustrating, in Figure 7 the distribution over the links of the backbone graph of the maximum detection time in the deterministic R-Tree scheduler, $\mathsf{M}_t[e]$, and the 99th percentile line for the memoryless schedulers (elapsed time to detection in 99% of the time). Figure 8 shows the same data for the schedulers RT-S LP and RT-S CON which were derived after restricting the set of paths over which optimization was performed. One can see that when there are strict requirements on worst-case detection times, deterministic schedules dominate.

MAX_e in memoryless schedulers:

algorithm	GN-U	GN-P	GN-Z	Clos
Convex	221.53	21.81	6.85	32.02
LP	132.05	12.65	2.67	32.02
Uniform	2787	12.73	249.28	34.00
SAMP SC	143.00	53.65	72	32.00
SAMP KT	243.00	22.74	72	32.00

$\mathsf{M}_e\mathsf{E}_t$ in deterministic schedulers

algorithm	GN-U	GN-P	GN-Z	Clos
SC	72.00	43.41	48.17	16.50
KT	122.00	11.97	4.28	16.50
RT CON	162.00	20.15	5.31	40.61
RT LP	173.90	18.92	2.91	40.53
RT-S CON	92.50	22.15	4.90	
RT-S LP	71.50	22.16	1.90	

$\mathsf{E}_t\mathsf{M}_e$ in deterministic schedulers

algorithm	GN-U	GN-P	GN-Z	Clos
SC	142.99	54.02	50.80	32.00
KT	234.30	34.02	4.54	32.00
RT CON	345.85	55.26	6.12	147.71
RT LP	531.12	65.31	7.05	156.80
RT-S CON	182.78	42.69	6.01	
RT-S LP	142.00	43.22	3.21	

$\mathsf{M}_e\mathsf{M}_t$ in deterministic schedulers

algorithm	GN-U	GN-P	GN-Z	Clos
SC	143.00	95.02	113.00	32.00
KT	243.00	35.65	9.00	32.00
RT CON	468.00	85.72	24.00	257.00
RT LP	833.00	97.79	13.95	225.00
RT-S CON	184.00	50.00	14.00	
RT-S LP	142.00	54.00	5.00	

Table 3: MAX_e objectives. Table shows expected time with memoryless schedules (same for all MAX_e objectives) and $\mathsf{M}_e\mathsf{E}_t \leq \mathsf{E}_t\mathsf{M}_e \leq \mathsf{M}_e\mathsf{M}_t$ on different deterministic schedulers.

9 Related work

This basic formulation of failure detection via probes applies in multiple network scales, from back bones to data centers [13, 11]. A recent application is testing of all forwarding rules in a software-defined network [12]. Beyond the detection of network failures, the fundamental optimization problems we study models classic and emerging resource replication and capacity allocation problems.

Previous considerations of the detection problem for network failures focused on MAX_e objective when all elements have equal importance (uniform priorities) [13, 11, 12]. In this particular case, deterministic scheduling is equivalent to finding a minimum size set of tests which covers all elements, which is the classic set covering problem. The optimal memoryless schedule is a solution of a simplified LP, which computes an optimal fractional cover. In practice, however, some elements are much more critical than others, and the uniform modeling does not capture that. Ideally, we would like to specify different detection-time targets for failures which depend on the criticality of the element. A set cover based deterministic schedule, however, may perform poorly when elements have different priorities and there was no efficient algorithm for constructing good deterministic schedules. Moreover, the SUM_e objectives, which were not previously considered for network failure detection application, constitute a natural global objective for overall performance, for example, when elements have associated fail probability, SUM_e minimization corresponds to minimizing expected failure detection time.

The special case of *singletons* (each test contains a single element) received considerable attention and models several important problems. The SUM_e objective on memoryless schedules is the subject of Kleinrock's well know "square root law" [9]. Scheduling for Teletext [2] and broadcast disks [1], can be formulated as deterministic scheduling of singletons. Both $\mathsf{E}_e\mathsf{E}_t$ and $\mathsf{M}_e\mathsf{M}_t$ objectives were considered. Our Kuhn-Tucker inspired scheduling for MAX_e generalizes a classic algorithm for singletons [7, 3] which has

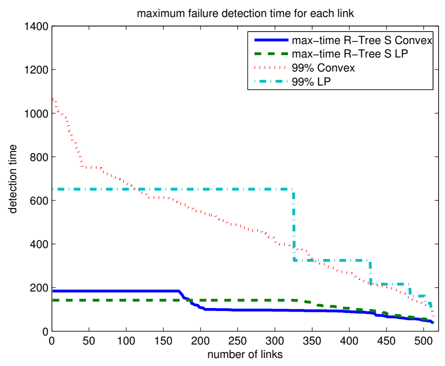


Figure 8: Distribution of time to detect a fault: RT-S LP and RT-S CON (deterministic) vs. memoryless over GN-U.

a factor 2 approximation for the $\mathsf{E}_e\mathsf{E}_t$ [3]. Bar-Noy et al. [3, 4] established a gap ≤ 2 between the optimal deterministic and memoryless schedules, this is in contrast to the difficulty of general subset tests, where we show that gaps can be asymptotic. Interestingly, however, even for singletons, $\mathsf{M}_e\mathsf{M}_t$ optimal deterministic scheduling is NP hard [3]. several approximation algorithms were proposed for deterministic scheduling [8, 3, 4]. In particular, Bar-Noy et al. [3, 4] proposed tree-schedules, which are an ingredient in our R-Tree schedule constructions, as a representation of deterministic schedules. Memoryless schedules with respect to the SUM_e objective modeled replication or distribution of copies of resources geared to optimize the success probabilities or search times in unstructured p2p networks [5]. Our convex program formulation extends the solution to a natural situation where each test (resource) is applicable to multiple elements (requests).

Conclusion

We conducted a comprehensive and unified study of the modeling, algorithmics, and complexity of probe scheduling. We reveal inherent gaps between different objectives and between stochastic and deterministic schedules and propose efficient scheduling algorithm with analytic performance guarantees. We demonstrate the value of this work by simulations on realistic networks. Beyond silent failure detection, we believe the optimization problems we address and our scheduling algorithms will find applications in other resource allocation domains.

References

[1] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. Broadcast disks: data management for asymmetric communication environments. In *ACM SIGMOD*, 1995.

[2] M. Ammar and J. Wong. On the optimality of cyclic transmission in teletext systems. *IEEE Tran. Communication*, 35(1):68–73, 1987.

[3] A. Bar-Noy, R. Bhatia, J. Naor, and B. Schieber. Minimizing service and operation costs of periodic scheduling. *Math. Oper. Res.*, 27(3):518–544, 2002.

[4] A. Bar-Noy, V. Dreizin, and B. Patt-Shamir. Efficient algorithms for periodic scheduling. *Computer Networks*, 45(2):155–173, 2004.

[5] E. Cohen and S. Shenker. Replication strategies in unstructured peer-to-peer networks. In *Proceedings of the ACM SIGCOMM'02 Conference*, 2002.

[6] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45:634–652, 1998.

[7] S. Hameed and N. H. Vaidya. Log-time algorithms for scheduling single and multiple channel data broadcast. In *Proc. of ACM/IEEE MobiCom*, 1997.

[8] C. Kenyon, N. Schabanel, and N. E. Young. Polynomial-time approximation scheme for data broadcast. In *ACM STOC*, 2000.

[9] L. Kleinrock. *Queueing Systems, Volume II: Computer Applications*. Wiley-Interscience, New York, 1976.

[10] R. R. Kompella, J. Yates, A. G. Greenberg, and A. C. Snoeren. Detection and localization of network black holes. In *INFOCOM*, 2007.

[11] H. X. Nguyen, R. Teixeira, P. Thiran, and C. Diot. Minimizing probing cost for detecting interface failures: Algorithms and scalability analysis. In *INFOCOM*, 2009.

[12] H. Zeng, P. Kazemian, G. Varghese, and N. McKeon. Automatic test packet generation. In *CONEXT*, 2012.

[13] Q. Zheng and G. Cao. Minimizing probing cost and achieving identifiability in probe based network link monitoring. *IEEE Tran. Computers*, 2012. to appear.

A Deferred Proofs

Proof of Lemma 2.2

Proof. We replace the original stochastic schedule by a cyclic stochastic schedule $\boldsymbol{\sigma}$-N by choosing some N and after each repetition of N time steps invoking the original schedule from time 0. We then take a new schedule $\boldsymbol{\sigma}'$-N which uniformly executes one of the N possible shifts of the cyclic schedule $\boldsymbol{\sigma}$-N. For each element e and for all t, $\mathsf{T}_{\boldsymbol{\sigma}'\text{-N}}(e,t) = \mathsf{E}_t[e|\boldsymbol{\sigma}\text{-N}]$, that is, the detection time at any time t on the combined shifted schedule is equal to the expected (over time) detection time on the cyclic schedule.

It remains to show that for any δ we can choose an N so that for all elements $\mathsf{E}_t[e]$ on the cyclic schedule is within $(1+\delta)$ of $\mathsf{E}_t[e]$ on the original schedule , that is, $\mathsf{E}_t[e|\boldsymbol{\sigma}\text{-N}] \leq (1+\delta)\mathsf{E}_t[e|\boldsymbol{\sigma}]$.

For each e, there must be N_e so that for all $h \geq N_e$,

$$(1/h)\sum_{t=1}^{h} \mathrm{T}(e,t) \leq \mathsf{E}_t[e](1+\epsilon) \ .$$

We take N to be the maximum of $\max_e N_e$, $\max_e \mathsf{T}(e,1)/\epsilon$, and $\max_e \mathsf{E}_t[e]/(|E|\epsilon)$.

For each element e, we need to bound the potential increase in expectation over $(1/N)\sum_{t=1}^{N} T(e,t)$ that is due to the "wrap around". Formally, bound $\sum_{t=1}^{N} \mathrm{T}_{\boldsymbol{\sigma}\text{-N}}(e,t)$ in terms of $\sum_{t=1}^{N} \mathrm{T}_{\boldsymbol{\sigma}}(e,t)$. The effect of the wrap around is at most that of replacing $\mathrm{T}(e,t)$ by $\mathrm{T}(e,t) + \Pr[\mathrm{T}(e,t) > N-t]T(e,1)$. $\sum_{t=1}^{N} \Pr[T(e,t) > N-t](N-t) \leq \mathsf{E}_t[e](1+\epsilon)$ and therefore $\sum_{t=1}^{N} \Pr[T(e,t) > N-t] \leq (1+\epsilon)\mathsf{E}_t[e]$. Hence, $\sum_{t=1}^{N} T(e,t)$ under the wrap around is at most $\sum_{t=1}^{N} T(e,t)$ in the original sequence plus $T(e,1)\mathsf{E}_t[e] \leq \epsilon N \mathsf{E}_t[e]$. Thus $(1/N)\sum_{t=1}^{N} T(e,t)$ under the wrap around is at most that of the original sequence plus $\epsilon\mathsf{E}_t[e]$. □

Proof of Lemma 2.4

Proof. Consider a stochastic schedule $\boldsymbol{\sigma}$. $\mathsf{E}_t[e]$ is well defined for all e and so does the $\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] = X$ The stochastic schedule is a distribution over deterministic schedules, but a technicality we need to circumvent is that the $\mathsf{E}_e\mathsf{E}_t$ may not be defined on some of these schedules even when defined for $\boldsymbol{\sigma}$. For $N \geq 1$, we can obtain a cyclic stochastic schedule by executing our schedule for the first N time steps, we then discard the sequence if there is an element e that is not probed even once, and repeat from the beginning. When we get a successful length N sequence, we cycle through it to obtain an infinite schedule. We show that for each $\delta > 0$, we can always choose N so that the $\mathsf{E}_e\mathsf{E}_t$ of the cyclic schedule is at most $(1+\delta)X$. This cyclic stochastic schedule is a distribution over cyclic deterministic schedules. Since on every cyclic schedule the $\mathsf{E}_e\mathsf{E}_t$ is well defined, the $\mathsf{E}_e\mathsf{E}_t$ of these schedules and also their expected $\mathsf{E}_e\mathsf{E}_t$, which is equal to the $\mathsf{E}_e\mathsf{E}_t$ of the cyclic stochastic schedule, is well defined. Therefore, there must be a deterministic cyclic schedule with $\mathsf{E}_e\mathsf{E}_t$ that is at most that of the cyclic stochastic schedule which in turn is at most $(1+\delta)X$, which establishes (6).

It remains to outline how to choose N. For each e, there must be N_e so that for all $h \geq N_e$ $(1/h)\sum_{t=1}^{h} T(e,t) \leq \mathsf{E}_t[e] + \epsilon$. We take N to be the maximum of $\max_e N_e$, $\max_e T(e,1)/\epsilon$, and $\max_e \mathsf{E}_t[e]/(n\epsilon)$.

We first observe that the probability that a sequence is discarded is at most ϵ. So by discarding these sequences we can not increase the expectation by more than $1/(1-\epsilon) \approx (1+\epsilon)$. Secondly, we need to bound the potential increase in expectation that is due to the "wrap around". This is handled as in the proof of Lemma 2.2.

We now establish the inequalities. Given a deterministic schedule and ϵ, we construct a cyclic deterministic schedule on which the $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{M}_t\mathsf{E}_e$ are within $(1+\epsilon)$ of the original deterministic schedule. We take $N_\epsilon \gg \mathsf{E}_e\mathsf{M}_t/\epsilon$ and such that the $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{M}_t\mathsf{E}_e$ on the first h time steps for all $h \geq N_\epsilon$ are at most $(1+\epsilon/2)$ times that of the original sequence. We now look at the "state", which is the last time since each edge was tested and time for next probe. There is a bound L such that the state at most time steps has all times at most L. We can now find a subsequence of the schedule that starts and finishes at the same state and that the objectives over it are at most $(1+\epsilon)$ times those of the original schedule. We turn it into a cyclic schedule.

We now take this cyclic schedule and construct a stochastic schedule $\boldsymbol{\sigma}'$ by selecting a start point uniformly at random. For each element e, we have

$$\mathsf{M}_t[e|\boldsymbol{\sigma}'] \leq \frac{\mathsf{M}_t[e|\boldsymbol{\sigma}]+1}{2} \,.$$

By combining,

$$\text{opt-}\mathsf{E}_e\mathsf{M}_t \leq \sum_e p_e \mathsf{M}_t[e|\boldsymbol{\sigma}'] \leq \sum_e \frac{\mathsf{M}_t[e|\boldsymbol{\sigma}]+1}{2} = (\mathsf{E}_e\mathsf{M}_t + 1)/2 \,.$$

By taking the minimum $\mathsf{E}_e\mathsf{M}_t$ over all deterministic schedules we conclude the claim. The argument for $\mathsf{M}_e\mathsf{M}_t$ is similar. □

Proof of Lemma 2.3

Proof. We use a reduction to exact cover by sets of size 3 (X3C). We obtain a scheduling instance using the same set of elements and subsets (tests) as the X3C instance. We use a uniform $\boldsymbol{p}$ over elements with $p_e = 1/(3k)$ for SUM$_e$ objectives and $p_e = 1$ for MAX$_e$ objectives.

We first consider deterministic schedules. From an exact cover, we define a deterministic schedule by cycling through the same permutation of the cover. The deterministic schedule has $\mathsf{M}_t[e] = k$ and $\mathsf{E}_t[e] = (k+1)/2$ for all elements e. The maximum $\max_e T(e,t)$ at any time t is k and the average is $(k+1)/2$. Therefore, the schedule has $\mathsf{M}_e\mathsf{M}_t$, $\mathsf{E}_t\mathsf{M}_e$, and $\mathsf{E}_e\mathsf{M}_t$ equal to k and $\mathsf{M}_e\mathsf{E}_t$, $\mathsf{E}_e\mathsf{E}_t$, and $\mathsf{M}_t\mathsf{E}_e$ equal to $(k+1)/2$.

Consider an arbitrary deterministic schedule and time t. We must have $\max_e T(e,t) \geq k$, since at most $3i$ elements can be covered in i probes, so to cover all $3k$ elements we need at least k probes. We have equality if and only if the sequence of k probes following t constitutes a cover. A cover of size k must be an exact cover. Therefore $\mathsf{E}_t\mathsf{M}_e = k$ implies exact cover of size k.

Similarly, we claim that on any schedule, $(1/k)\sum_e T(e,t) \geq (k+1)/2$. This is because $\sum_e T(e,t) = \sum_e m_e$, where m_e is the smallest d such that $e \in \sigma_{t+d}$. Since there can be at most 3 elements of each value of $m_e \geq 1$, we have that $\sum_e T(e,t) \geq 3\sum_{d=1}^{k} d = 3k(k+1)/2$ and our claim follows. Moreover, equality holds only if the sequence of k probes from t on is an exact cover. Therefore $\mathsf{M}_t\mathsf{E}_e = (k+1)/2$ implies exact cover of size k.

Consider an arbitrary deterministic schedule and let q_e be the average probing frequency of element e (we assume convergence). We have $\mathsf{M}_t[e] \geq 1/q_e$ and $\mathsf{E}_t[e] \geq (1+1/q_e)/2$. Moreover, equality can hold only when $1/q_e$ is intergral and probes are evenly spaced every $1/q_e$ probes except for vanishingly small fraction of times. For the X3C instance we have $\sum_e q_e = 3$, and from convexity, $\sum_e 1/q_e$ or $\max_e 1/q_e$ are minimized only when all q_e are equal to $1/k$. This means that the $\mathsf{M}_e\mathsf{M}_t$ and $\mathsf{E}_e\mathsf{M}_t$ can be equal to k or the $\mathsf{M}_e\mathsf{E}_t$ and $\mathsf{E}_e\mathsf{E}_t$ are equal to $(k+1)/2$ equal to $(k+1)/2$ only if each element is probed every k probes (except vanishingly small) number of times. This means that most sequences of k consecutive probes constitute an exact cover.

We now consider stochastic schedules. From an exact cover, we define a stochastic schedule by a uniform distribution $(1/k)$ on each of the k shifts of the same permutation of the cover. On this schedule, all our objectives have value $(k+1)/2$. It remains to show that for each of the objectives, a schedule with time $(k+1)/2$ implies an exact cover.

Observe that with our choice of weighting, on any schedule opt-MAX$_e \geq$ opt-SUM$_e$. Therefore if the stochastic optimum of either the SUM$_e$ or MAX$_e$ objectives is $(k+1)/2$, then opt-SUM$_e$ is also $(k+1)/2$ which implies, from (6), that opt$_D$-$\mathsf{E}_e\mathsf{E}_t$$= (k+1)/2$, which implies exact cover. $\square$

Proof of Lemma 4.2

Proof. We choose n, $\ell \geq 1$, and $m \geq 2\ell$ such that $n = \binom{m}{\ell}$, and construct an instance with n elements and m tests such that each element is included in exactly ℓ test and every subset of ℓ tests has exactly one common element. We use a unifrom $\boldsymbol{p}$.

The instance is symmetric and therefore in the solution of the convex program (9) or (10) all the m tests have equal rates $q = 1/m$. The memoryless schedule with this $\boldsymbol{q}$ optimizes both SUM$_e$ and MAX$_e$ for $\boldsymbol{p}$. For any element and any time, the expected detection time by a memoryless schedule with $\boldsymbol{q}$ is m/ℓ. But for any particular deterministic schedule and a particular time there is an element that requires $m - \ell$ probes (for any sequence of $m - \ell$ tests there must be at least ℓ tests not included and we take the element in the intersection of these tests. This means that at any time, the worst-case element detection time is factor $\frac{m-\ell}{m/\ell} \geq \ell/2 = \Theta(\ln n/\ln m)$ larger than the memoryless optimum.

When fixing the number of tests m, this is maximized (Sperner's Theorem) with $\ell = m/2$ and the MAX$_e$ ratio is $\Theta(m)$. When fixing the number of elements n, the maximum ratio is $\arg\max_\ell n = \binom{2\ell}{\ell}$ and we obtain $\ell = \Theta(\ln n)$.

We use the same construction as in Lemma 4.2 and take a uniform $\boldsymbol{p}$ over elements. Lastly, to show $\mathsf{E}_e\mathsf{M}_t$ of $\Omega(\ln \ell)$ consider a sequence of m probes. The expectation over elements of the number of probes that test the element, is at most ℓ. So at least half the elements are probed at most ℓ times. There are at least $m/2$ distinct tests. The expected over e maximum difference between probes to an element e over a sequence of m is $\Omega(\ln \ell)m/\ell$. This is because every combination of ℓ distinct probes corresponds to an element, and thus, for the "average" element, the probes can be viewed as randomly placed, making the expectation of the maximum interval a logarithmic factor larger than the expectation.

We now show how the instances can be realized on a network. We use n pairs of links. Each pair includes a link which corresponds to an element in our instance and a "dummy" link. The pairs are connected on a path of size n. Each test is an end to end path which traverses one link from each pair. Each (real) link is covered by exactly ℓ paths and every subset of ℓ paths has one common (real) link. The network is a path of length n of pairs of parallel links, a real and a dummy link. Real links e have $p_e = 1$ and the dummy link have $p_e = +\infty$. (If we want to work with respect to some SUM$_e$ optimum we take $p_e \equiv p$ (for some $p < 1$) for real links and $p_e = 0$ for dummy links). Each path traverses one link from each pair and includes ℓ real links. $\square$

The Lemma is tight in the sense that it is always possible to get a schedule with D2M equal to m by cycling over a permutation of the tests.